

# Model-Driven Architecture<sup>®</sup> and the Semantics of Business Vocabulary and Business Rules

Stan Hendryx, Hendryx & Associates  
[stan@hendryxassoc.com]  
September 26, 2005

---

## Abstract

Language is the main medium of social and technical constructivism.<sup>1</sup> The *Semantics of Business Vocabulary and Business Rules* specification (SBVR) recently approved by the Object Management Group is intended to be the basis for a formal and detailed natural language declarative description of a complex entity, such as a business. Such formal vocabularies and descriptions can be interpreted and used by computer systems. SBVR is expected to become an integral part of the OMG's Model-Driven Architecture<sup>®</sup> (MDA<sup>®</sup>). SBVR is already included or expected to be included in several emerging standards, including specifications for business process modeling, and is being taken up by some high-profile research and development projects. This paper introduces SBVR by using its concepts to provide an overview description of MDA itself. The paper is aimed at MDA decision makers and modelers, and people who want to understand more about formalizing business vocabularies. Conceptualization and representation play fundamental roles in thinking, communicating, and modeling. For each concept there is a triad of 1) the concept in our minds, 2) the real-world things conceptualized by the concept, and 3) a representation of the concept that we can use to think and communicate about the concept and its corresponding real-world things. A conceptual model is a formal structure representing a possible world, comprising a conceptual schema and a set of facts that instantiate the conceptual schema. The conceptual schema is a combination of concepts and facts of what is possible, necessary, permissible, and obligatory in each possible world. The set of facts instantiates the conceptual schema by assertion to describe one possible world. A rule is a fact that asserts either a logical necessity or an obligation. Obligations are not necessarily satisfied by the facts; necessities are always satisfied. 'Meta level' is a concept used in MDA to classify concepts based on whether their instances are individuals or other concepts. A modeling language is defined as a conceptual model whose facts assert concepts. The abstract syntax of a modeling language is the set of facts in its definition and appears as the conceptual schema in a use of the language for a new conceptual model. Different modeling languages are distinguished by the generality or specificity of the concepts they assert. A summary of MDA modeling languages is presented. MDA model transformations are a succession of mappings of representations from the point of origin, the business model, to their ultimate representation in the schema of an information system. An information system is represented by a conceptual model whose facts assert individuals, its conceptual schema being represented by the database schema, object model and business rule enforcement procedures of the system, and its facts represented by the data in the system. Information systems enforce business rules through operation and use of the system in business processes. System transactions change the data, thereby changing the underlying conceptual model represented by the system to be a model of a different possible world of the modeled business.

---

<sup>1</sup> Paolo Dini, London School of Economics, private communication.

## Introduction

The Object Management Group (OMG) recently approved the *Semantics of Business Vocabulary and Business Rules* (SBVR) to become a final adopted specification of the OMG. SBVR is a landmark for the OMG, the first OMG specification to incorporate the formal use of natural language in modeling and the first to provide explicitly a model of formal logic. Based on a fusion of linguistics, logic, and computer science, and two years in preparation, SBVR provides a way to capture specifications in natural language and represent them in formal logic so they can be machine-processed. Hendryx & Associates lead the organization of the cross-functional team that created SBVR, and is a co-submitter of the specification. SBVR is expected to become an integral part of the OMG's Model-Driven Architecture<sup>®</sup> (MDA<sup>®</sup>). This paper introduces SBVR as part of MDA by explaining MDA in terms of SBVR. The paper is intended for people who are deciding about using MDA and formal modeling in their projects, modelers and MDA practitioners, and those who want to understand more about SBVR and how it fits into MDA.

To be most effective, the MDA modeling cycle needs to begin at the beginning, with business and business people. This is where business models, including business process models, are defined and system requirements are specified. Business models are at the point of origin, where motivations for a business are apparent and where business vocabulary and business rules are defined. Business vocabulary and business rules permeate the organization and all of its processes and information systems, at every level of detail. If business vocabulary and business rules can be captured formally at the point of origin, they can be productively reused in processes and systems throughout the business, to coordinate and integrate related parts of the business, making each part best serve the whole. Business processes derive their vocabulary for events, actors, and actions from business sources. Business processes enforce business rules and make them operational. Indeed, the motive of business processes is to enforce business rules. Business processes progress according to business rules. Making business vocabulary and business rules available to business process designers increases both their efficiency and the effectiveness of their process designs. Externalizing the controlling vocabulary and rules from process definitions and information system designs allows these shareable elements to be shared and to be positioned as first-class entities of the business, where they can be managed by business people and their impacts can be assessed.

The importance of capturing business vocabulary and business rules at their point of origin creates a pressing need for formal methods to capture requirements in natural language. This is where they are best authored and validated by business experts, in a way that allows for their subsequent incorporation into the system development automation stream. A role of SBVR in MDA is to fill this need. SBVR contains a vocabulary for conceptual modeling and captures expressions based on this vocabulary as formal logic structures. The SBVR vocabulary allows one to specify formally representations of concepts, definitions, instances, and rules of any knowledge domain in natural language, including tabular forms. These features make SBVR well suited for describing business domains and requirements for business processes and information systems to implement business models. Several MDA-related OMG works in progress are expected to incorporate SBVR, including

- Business Process Definition Metamodel (BPDMM)
- Organization Structure Metamodel (OSM)
- Business Motivation Model (BMM)
- UML Profile for Production Rule Representation (PRR)
- UML Profile for the Department of Defense Architecture Framework/Ministry of Defense (Canada) Architecture Framework (DoDAF/MODAF).

The Ontology Definition Metamodel (ODM) has been made compatible with SBVR, primarily by aligning the logic grounding of the ISO Common Logic specification (CL) referenced by ODM with the SBVR Logical Formulation of Semantics vocabulary. CL itself was modified specifically so it potentially can include the modal sentence requirements of SBVR. ODM provides a bridge to link SBVR to the Ontology Web Language for Services (OWL-S), Resource Description Framework Schema (RDFS), Unified Modeling Language (UML), Topic Maps (TM), Entity Relationship Modeling (ER), Description Logics (DL), and CL. Other programs outside the OMG are adopting SBVR. The Digital Business Ecosystem (DBE), an integrated project of the European Commission Framework Programme 6, has adopted SBVR as the basis for its Business Modeling Language. The World Wide Web Consortium (W3C) is assessing SBVR for use in the Semantic Web, through the bridge provided by ODM. SBVR will extend the capability of MDA in all these areas.

MDA is a technical process for system development based on formal modeling. As such, the internals of MDA are not easily explained to people not involved with the technology. SBVR provides a set of concepts to describe complex things formally, concepts most educated people, technical or not, are probably already familiar with or that they can readily assimilate if they are not. The idea behind this paper is to provide an explanation of MDA fundamentals using concepts of SBVR. The goal was to provide an explanation that is close but not overly technical, in plain language. In this way, it is hoped the reader will gain both a better understanding of modeling and MDA, and insights into the workings of SBVR and how SBVR works with MDA. Care has been taken to define MDA technical terms in plain language, in terms of SBVR. In SBVR, the normative concepts and the terms that represent these concepts were deliberately chosen to be as close to the ordinary dictionary meaning of the terms as possible. Terms found throughout this paper in **boldface** have definitions in the *Semantics of Business Vocabulary and Business Rules* specification, OMG document *bei/05-08-01*, often recited herein. The use of singular construction prevalent in the paper is deliberate and is also characteristic of SBVR expressions.

The paper is organized as follows. Conceptualization and representation, which are the foundations for thinking and communicating, are discussed first, to establish the main ideas of formal modeling. Relationship of concepts to instances is described, and the notion of meta levels is introduced. With this background, ‘conceptual model’ is carefully defined, followed by a discussion of the meaning of rules and facts in a conceptual model. A categorization scheme for conceptual models is introduced, based on meta levels. Modeling languages are shown to be a particular category of conceptual model in which the existence of concepts is asserted in the set of facts of a conceptual model. The distinction is made between a definition of a language and a use of the language; crossing these distinctions often leads to a great deal of confusion in discussions about modeling. Limitations of the inherent semantics of a modeling language to join models semantically is discussed. The concept of ‘reflection’ is introduced and its importance to MDA explained, as the way to discover facts about instances. The idea of general- and special-purpose modeling languages is introduced, and the trade-off between expressivity and ease of use is discussed. The need for reference schemes to identify individuals in models is explained. How models are incorporated into information systems is discussed, and the effect of information system transactions on the underlying model is explained. The article concludes by providing a summary of MDA general- and special-purpose languages, and an overview of MDA model transformations in getting from business model to running information systems.

#### Conceptualization and Representation

“MDA is an approach to system development...[that]... provides a means for using models to direct the course of understanding, design, construction, deployment, operation, maintenance and modification.” [MDA Guide *omg/03-06-01*]

MDA modeling is a formal approach for carefully describing a part of the world and systematically translating the description from English or other natural language through a succession of representations in various MDA modeling languages and programming languages to a representation that is held in an information system. To understand MDA, it is instructive to begin by reviewing how language works as we think and communicate about the world.

Consider first the concept ‘**concept**’:

**concept**: unit of knowledge created by a unique combination of characteristics

A **concept** exists only in the mind of a person who is thinking about it. A particular concept corresponds to a **thing** or **things** in the world. By ‘thing’ is meant anything perceivable or conceivable. A thing is anything we can perceive through our senses (or convert to sensations with instruments) or anything we can think about. A thing can be either physical or abstract. A concept can correspond to things that are unitary (the concept ‘automobile’) or to things that involve related unitary things (the concept ‘person owns automobile’). A concept of a unitary thing is called a ‘**noun concept**’ in SBVR. A concept of a relation involving noun concepts is called a ‘**fact type**’<sup>2</sup>. Which things correspond to a particular concept are determined by the **definition** of the concept.

We often give a name to a concept by associating a **symbol** with the concept, such as a written or spoken word or phrase or icon. We use this symbol as a **representation**<sup>3</sup> of the concept when we communicate about the concept through speech or writing. For example, we see a thing that has the **characteristics** of being a road vehicle with four wheels, powered by an internal-combustion engine, and able to carry a small number of people, and we conceptualize it as a certain kind of thing. We use the symbol ‘automobile’ to represent the concept (in English), and we say, “that is an automobile.” Thus, for each concept there is a triad of 1) the concept in our minds, 2) the real-world things conceptualized by the concept, and 3) a representation of the concept that we can use to think and communicate about the concept. Note that a concept may have many different representations, in the same language or different languages. Agreement among communicating parties on the alignment between representations and concepts is a prerequisite to effective communication.

People with whom we want to communicate about automobiles must also associate the symbol ‘automobile’ with the same concept we have in mind, or there will be a miscommunication. Communicating parties must share the concept, in their minds, and use a shared representation of it in their communication, in order to understand and to be understood. Machines are often used as intermediaries in communication or surrogates for people in interactions with other people. Machines do not hold concepts, because a machine does not have a mind. However, communicating machines must hold semantically equivalent representations of the concepts and must manipulate each representation in relation to the representations of other concepts. The machine representations and manipulations must be consistent with how people understand the concepts and how they understand that the things conceptualized by the concepts are manipulated in the real world. At the interfaces between people and machines, the representations must be correctly understandable by people. IT development involves defining machine representations for concepts and defining interactions of the representations that parallel the real world. Specifying machine representations and their interactions

---

<sup>2</sup> A mention of a concept is denoted by enclosing its designation in single quotes, ‘fact type’. A use of a concept in a sentence is unquoted. Crossing use and mention in formal models can create illogical constructs and apparent paradoxes.

<sup>3</sup> A **representation** is a portrayal of a meaning by an expression, which is something that expresses or communicates, but that is independent of its interpretation, e.g. the sequence of characters ‘automobile’.

that correctly correspond to the world is where modeling and MDA come in. SBVR provides a natural starting point for the MDA process, by using the natural language that is the basis for thought and person-to-person communications for modeling what people have in mind. With SBVR, the language does not get in the way of domain experts modeling their domain; they are not required to think about implementations of their ideas or coded machine representations. With SBVR, they get to focus on expressing their ideas, as complex as they may be.

#### Extensions and Instances

Some concepts may have only one thing in the world that corresponds to each of them (e.g. The Statue of Liberty or Darth Vader). Some other concepts may have many things that correspond to each of them (e.g. automobile, unicorn). The set of all things that are conceptualized by a particular concept is called the **extension** of the concept. For example, the extension of the concept 'automobile' is the set of all automobiles. In SBVR, this set can be specified to include only automobiles that are explicitly declared in the domain model, or it can be left open to include any automobile at all, depending on the intended meaning in a particular model.

An individual thing that is in the extension of a concept is called an **instance** of the concept. In object-oriented analysis and programming, an 'object' is a representation of an instance. Sometimes the term 'instance' is used to refer to a representation of an instance, which can cause confusion.

#### Meta Levels

The instances of some concepts are individual things, such as a particular automobile, which is an instance of the concept 'automobile'. The instances of other concepts are themselves concepts, such as the concept 'type of vehicle', of which the concept 'automobile' is an instance. Concepts whose instances are individuals are known as 'first order concepts'. Concepts whose instances are concepts are known as 'higher order concepts'. There is no theoretical limit as to the number of conceptual levels that a concept can be removed from an individual. For example, the taxonomy of zoology has seven conceptual levels: kingdom, phylum, class, order, family, genus, species. An individual creature is an instance of a species, e.g. a person is an instance of the species *Homo sapiens*; we say that a person is a *Homo sapiens*. A person is not an instance of any genus, but rather the concept that is the species '*Homo sapiens*' is an instance of the concept that is the genus 'Homo', and we say '*Homo sapiens* is a Homo'. In MDA, the number of conceptual levels that a concept is removed from individual is called the 'meta level' of the concept or its representation, often designated M-0 (individual), M-1, M-2, or M-3. As we shall see below, the distinction between first-order and higher-order plays an important role in the use of a model. Beyond the first-order/higher-order distinction, however, the absolute conceptual level of a concept is relatively unimportant.

#### Conceptual Models, Conceptual Schemas, and Possible Worlds

A **conceptual model** is a representation of an instance that describes the instance in terms of concepts and **facts**<sup>4</sup>. A conceptual model is thus a definition of an instance, which can be an arbitrarily complex thing, such as a business. Because of the assumed complexity of a modeled instance, the instance that is the subject of a conceptual model is often called a **possible world**. A conceptual model is a combination of a **conceptual schema** and a **set of facts**. The conceptual schema is a combination of concepts and facts of what is possible, necessary, permissible, and obligatory in each possible world. The conceptual schema represents the level of abstraction of a model; it is possible to model a thing at different levels of abstraction, i.e. to create different models of the thing that have different conceptual schemas. The set of facts asserts individuals that exist in a possible world, and their characteristics.

---

<sup>4</sup> A **fact** is a proposition that is asserted to be true. A **proposition** is the meaning that is asserted when a sentence is uttered or inscribed and which is true or false.

Each individual is an instance of some concept of the conceptual schema. Each fact asserts an instance of some concept of the conceptual schema. That is, the set of facts defines what is being modeled in terms of the conceptual schema. The extension of a conceptual schema is the set of all possible worlds that are consistent with the conceptual schema. The set of facts specifies one possible world from this set. The **actual world** is a special case of a possible world wherein each fact in the set of facts is an **actuality** – there is an instance in the actual world that corresponds to each fact.

#### Rules

Each **necessity** in the conceptual schema is satisfied by the conceptual model, but the **obligations** are not necessarily satisfied. ‘Necessity’ and ‘obligation’ are categories of ‘**rule**’. Necessities define structural relationships in each possible world, and are definitional in nature. Obligations define duties of a certain party or parties to do a certain act. Since people (and machines) do not always do what they are supposed to do, the obligations are not necessarily satisfied in a particular world (possibly including the actual world).

#### Facts of a Conceptual Model

There are two kinds of facts in the set of facts of a conceptual model. An *existential fact* is used simply to assert the existence of something, “There is an automobile that has license number ‘CA ABC-123’.” An *elementary fact* is an assertion that an instance has a property, “The automobile having license number ‘CA ABC-123’ is blue,” or that one or more instances participate in a relationship, “The automobile having license number ‘CA ABC-123’ is owned by the person named ‘Stan Hendryx’.” An elementary fact cannot be split into simpler facts with the same instances without information loss. An elementary fact may be treated as an instantiation of an irreducible relation, called a **fact type**, which is a concept whose instances are all facts, e.g. the fact type ‘person owns automobile’. Each fact type has at least one role. A **role** is a concept (other than a fact type) that corresponds to things based on their playing a part, assuming a function or being used in some **situation** being described by a fact, e.g. ‘person’ and ‘automobile’ are roles in the fact type ‘person owns automobile’. The number of roles in a fact type is called the ‘arity’ of the fact type. SBVR supports fact types of any arity, including unary (1 role), binary (2), ternary (3), or *n*-ary (*n* roles). Most elementary fact types have 3 or fewer roles.

A fact type may be objectified as a noun concept, e.g. the concept ‘automobile ownership’ objectifies the fact type ‘person owns automobile’. **Objectification** allows a fact type to serve as a noun concept in other fact types, and implies fact types relating the objectification to the original noun concepts, e.g. ‘automobile ownership’ implies ‘automobile ownership involves person’ and ‘automobile ownership involves automobile’<sup>5</sup>.

A **fact type reading** is a representation of a fact type. Multiple fact type readings of a fact type are typical. Different fact type readings use different verbs and possibly differ in their ordering of the roles, but are considered to express the same fact if they mean the same. Thus, ‘automobile is owned by person’ and ‘person owns automobile’ are two readings of the same fact type. This form of reading, being a sentence, is called a **sentential form**. ‘Person owning automobile’ and ‘automobile owned by person’ are **nominal restrictive forms** that refer to one of the participating roles or the other. They are also readings of the same fact type, because they refer to the same relation. Nominal restrictive forms are common in rules.

#### Categories of Conceptual Models

It is convenient to have a **categorization scheme** for conceptual models that is based on the kinds of concepts that appear in the conceptual schema of the conceptual model, whether the concepts are all

---

<sup>5</sup> Underlining is frequently used in SBVR to delimit a noun concept, which can have more than one word in its designation.

first-order concepts, or whether there are higher-order concepts in the conceptual schema, and if so, the range of possible higher-order concepts that is permitted by the conceptual schema.

A conceptual model whose conceptual schema contains only first-order concepts, that is, contains only concepts whose instances are individuals, is called in MDA a ‘M-0 model’ or ‘implementation’ or ‘data’. M-0 denotes that the set of facts asserts only individuals.

A conceptual model whose conceptual schema contains higher order concepts whose instances are first-order concepts is called in MDA a ‘M-1 model’ or simply, a ‘model’. MDA and the Meta Object Facility™, MOF™, also historically defined an M-2 model as a ‘metamodel’, and an M-3 model as a ‘metametamodel’, each level having a conceptual schema one conceptual level higher than the previous. In this archaic MDA scheme, the model designation refers to the meta-level of the set of facts of the model. The designations M-0, M-1, M-2, and M-3, and the former restriction to four meta levels, are being deprecated in MDA, based on the realization that there is no inherent limit on the number of levels and no need for MDA to distinguish categorically among higher-order models. It is still useful to distinguish M-0 from higher-order models.

#### Abstract Languages and Syntax

A language can be defined by a conceptual model. In MDA, a conceptual model whose set of facts asserts concepts is called an ‘abstract language’. For example, if a conceptual model asserts “There is an automobile” (concept of ‘automobile’ is introduced) the model would be defining an abstract language that includes ‘automobile’ in its vocabulary. Since such a model specifies a vocabulary, it is a model of a language, which is a method of communication consisting of the use of structured representations. The language is called ‘abstract’ because only the structural elements, not notation, of the language are specified in the language definition model, e.g. ‘Class’, ‘Association’, ‘Property’ (in the Unified Modeling Language™, UML™). The set of concepts asserted by the set of facts of the language model is called the ‘abstract syntax’ of the language. The abstract syntax includes facts that associate and constrain the vocabulary concepts it asserts. It is typical in MDA to specify the notation, or ‘concrete syntax’, separately from the abstract syntax of a language.

Separating the abstract and concrete syntax specifications of a language leaves open the possibility for alternative concrete syntaxes for the language. Concrete syntax can be a graphical notation convention, as in case of a graphical language like UML, or a textual grammar, as in the case of a textual language like SBVR. Of course, some concrete syntax is needed to express abstract syntax. In MDA, UML notation is used for this purpose for graphical models. The abstract syntax of SBVR is expressed in the SBVR specification itself using a notation called ‘SBVR Structured English’, which is a controlled subset of ordinary English grammar. The terms shown in **boldface** in this document represent concepts in the abstract syntax of SBVR; only a few of the 400+ concepts in the abstract syntax of SBVR are illustrated in this paper.

In MDA, one language can be defined more or less in terms of another, depending on the extent the language being defined refers to concepts in the language being used. Defining one abstract language in terms of another is called ‘meta modeling’ in MDA. A language definition is a conceptual model in which the language being used comprises the conceptual schema and the language being defined comprises the set of facts. The language being used is the ‘meta-language’ of the language being defined. It is very important when discussing abstract languages to keep clearly in mind the point of view of the discussion, whether the *definition* of the language is being discussed, or whether a *use* of the language is being discussed. When the definition of a language is discussed the language is in the set of facts of the definition. When a use of a language is being discussed, the language is in the conceptual schema of the use. Much confusion and miscommunication results when these points of view are

crossed. Such confusion is so common that there is an informal term for it in the OMG: ‘meta-model-muddle’. MOF is the common meta-language among all MDA languages. MOF is also the meta-language of choice for the object-oriented MDA languages, particularly UML and its derivatives. SBVR is the meta-language of choice for natural language models in MDA. There is a MOF model of SBVR so that SBVR models can be structurally linked at the level of individual facts with other MDA models based on MOF. SBVR models can be semantically linked to other models through vocabulary.

What, one might ask, is the meta-language of MOF and of SBVR? It turns out each of these is its own meta-language. MOF and SBVR are each defined in terms of itself. In order for a language to be able to describe itself, it must be a ‘reflective language’. That is, there must be some category of ‘concept’ in the language that appears (is ‘reflected’) in its own meta-language.

#### Reflection

An abstract language that includes the fact type ‘**concept has instance**’ in its abstract syntax, or equivalently ‘instance is of concept’, allows discovery of information about the instance. The ability of an instance to discover information about its own structure and context is called ‘introspection’. Having introspective capabilities is called (objectified) as ‘reflection’ in MDA languages, and a language that has this feature is called a ‘reflective language’. Reflection is vital to MDA, since transformations need to be able to determine the concept(s) and characteristics of an instance. Reflection is also important in forming and answering conceptual queries, such as, “What automobile is owned by what person?” A reflective language can be defined in terms of itself. MOF and SBVR are each a reflective language, and therefore each can be used for meta-modeling. Note that a reflective language must include in its vocabulary some category of the concept ‘concept’ and the concept ‘instance’. The category of ‘concept’ involved in MOF is ‘Class’, and an instance of Class is an Object. The fact type ‘Object is of Class’ is found in the package MOF::Reflection, in the form of the function ‘Object.getMetaClass()’. UML and other MDA languages that inherit from MOF inherit MOF::Reflection and have the capability for introspection. Similarly, languages that inherit from SBVR are reflective.

#### Reference Schemes

It is often necessary to identify uniquely an individual in a conceptual model. To accomplish identification, a set of fact types, called a **reference scheme**, can be specified for a concept such that an individual in the extension of the concept is uniquely identified by the facts asserted in the reference scheme of the concept. In the examples above, the reference scheme for ‘automobile’ is the set of fact types ‘license number is issued by state’ and ‘automobile has license number’, i.e. an automobile is uniquely identified by a state issuing the license number and the license number of the automobile, e.g. ‘CA ABC-123’. A concept can have multiple reference schemes, e.g. vehicle identification number (VIN) for automobile, in addition to issuing state and license number. Reference schemes should represent stable properties of instances so an instance can continue to be identified even if some of its other characteristics change. Parties must agree on the reference scheme used in communication about individuals so they can each correctly identify the individuals.

#### Trade-off between Expressivity and Ease of Use

Note that the more specific the meaning of a concept of an abstract language is to the meaning of a concept that a modeler wants to declare in the model, the fewer facts will be needed to complete the model. A ‘universal’ modeling language needs only ‘concept’ and ‘instance’ in its abstract syntax to model any thing, but then everything about it needs to be in the model, a laborious and exacting task. With more specific concepts in the abstract language, the expressivity is more restricted, i.e. the smaller is the set of possible worlds the language can describe, but the easier it is to describe them, the easiest being simply to instantiate abstract syntax concepts directly. This fundamental inverse relation between

expressivity and ease of use leads people to build special-purpose languages for domains they want to model extensively, to reduce the modeling effort and make their models easier to interpret by domain experts. The challenge is then to link the domains for communication across domains.

Frequently, models can be semantically linked only by their use of vocabulary, rather than through a shared conceptual schema. This is particularly true of models based on the general purpose languages and languages designed to define other languages. For example, the meaning of a UML model that only uses 'Class' and 'Association' is determined entirely by the vocabulary used to name model elements. It is impossible to relate two such models semantically except via the vocabularies chosen for names of model elements, other than possibly assigning a globally unique identifier (URI) to each element. This same problem applies to SBVR models as well, by design. SBVR assiduously avoids making ontological commitments to anything other than vocabularies needed to describe vocabularies, meaning, and logic. Semantically joining or merging SBVR models is intentionally designed to rely on the inherent compatibility of the vocabularies being joined, rather than providing any kind of a top-level or universal ontology within SBVR itself. Languages that are very specific and assert only first-order concepts can be connected semantically through a shared conceptual schema. Such languages are analogous to business forms that are filled out with specific information, or data input forms in a computer system, or database schemas that have common elements on which they can be joined. There is an awareness of the need for common top-level and basic vocabularies to serve as a semantic anchor on which to ground models that need to be semantically joined. Providing a semantic anchor at the level of the business vocabulary is particularly attractive, because such an anchor can help semantically integrate a whole business, or industry, or economy. Language is the main medium of social and technical constructivism. Work on developing such ontologies is growing among collaborating industry groups. Development of such an ontological anchor is currently outside the scope of MDA.

#### Transactions change a Model

An enterprise information system may hold a M-0 model of an enterprise. It is the aim of most enterprise information systems that the model held by the information system is of the actual world. An enterprise often expends considerable resources to maintain its actual world model. The conceptual schema of the M-0 model is represented by the database schema, the system's software object model, and the system's representations of business rule enforcement procedures. The set of facts is represented by the data in the database. The possible world represented by the conceptual model in the information system changes and becomes a different possible world each time a fact is added, negated, or deleted from the model. That is, each transaction of the enterprise produces a new conceptual model, representing a different possible world of the enterprise, and, objectively, the changing actual world. The conceptual schema can change, which changes the set of possible worlds of the enterprise model. A change to the conceptual schema may result in the facts also changing, since the necessities must always be satisfied. Change to the conceptual schema may result in some obligations not previously satisfied being satisfied (if rules were loosened), or vice versa (if rules were tightened).

#### MDA General Purpose Modeling Languages

An abstract language whose conceptual schema includes the concept 'concept' is considered a general purpose language, because it can be used to declare any concept. The only truly general purpose abstract language of MDA is SBVR; 'concept' in SBVR is unconstrained. The conceptual schemas of UML and MOF contain the concept 'Class' that is a category of 'concept' that is delimited from 'concept' by the characteristics and constraints specified in the respective language specifications. In these languages, an instance of Class declares the required and permitted structure of corresponding Objects in the possible worlds modeled by these languages, in terms of the specification of Class in the languages. Thus, UML and MOF are quasi-general purpose languages, subject to their object-oriented

constraints. EMOF (Essential MOF), a MOF dialect, is very simple, having only a few concepts in its abstract syntax, notably 'Class' and 'Property', and a few others. 'Association' is notably absent from EMOF. 'Association' in UML corresponds to a subtype of 'fact type' in SBVR. 'Property' in EMOF can model binary fact types, and only binary fact types, through the contrivance of two opposite Properties on what would be considered to be the ends of a binary association. CMOF (Complete MOF) adds 'Association' and some other concepts that make it easier to model databases and related languages. UML adds many other concepts to its abstract syntax, particularly some that make it easier to model behavior. The expressivity of each language, i.e. the possible worlds it can describe, is limited by the concepts it declares and the associations of these concepts that it permits.

MOF is designed to define other modeling languages in MDA and to serve as a hub in transformations between languages based on MOF. The two dialects of MOF, EMOF and CMOF, are designed to provide a lightweight language for relatively simple applications, and a more robust language for more demanding applications, respectively.

There is a core model that is common to MOF and UML, to facilitate transformations between models based on either UML or MOF. The core includes 'Class' and 'Property'. The UML is designed as a general purpose object-oriented modeling language. The UML has provisions for modeling object structures and for modeling behavior using any of a few different paradigms provided.

SBVR is for modeling in natural language. Based on linguistics and formal logic, SBVR provides a way to represent statements in controlled natural languages as logic structures called **semantic formulations**. SBVR is intended for expressing business **vocabulary** and **business rules**, and for specifying business requirements for information systems in natural language. SBVR models are declarative, not imperative or procedural. SBVR has the greatest expressivity of any OMG modeling language. The logics supported by SBVR are typed first order predicate logic with equality, restricted higher order logic (Henkin semantics), restricted deontic and alethic modal logic, set theory with bag comprehension, and mathematics. SBVR also includes projections, to support definitions and answers to queries, and questions, for formulating queries. Interpretation of SBVR semantic formulations is based on model theory. SBVR has a MOF model, so models can be structurally linked at the level of individual facts with other MDA models based on MOF.

#### MDA Special Purpose Modeling Languages

The OMG has defined several special purpose abstract languages in addition to UML, MOF, and SBVR. An OMG language that specializes UML is called a 'UML Profile'. A language that specializes MOF is called a 'metamodel' or a 'MOF model'. Some of these special purpose languages include the Common Warehouse Metamodel, and UML Profiles for CORBA®; CORBA Component Model (CCM); Enterprise Application Integration (EAI); Enterprise Distributed Object Computing (EDOC); Quality of Service and Fault Tolerance; Schedulability, Performance, and Time; and the UML Testing Profile. Other special purpose languages are works in progress in the OMG. Some of these refer only to parts of the UML metamodel, sometimes only 'Class'. Semantic linking in such cases is based on shared vocabulary.

#### Model Transformations

As discussed above, it is possible and convenient to conceptualize many different abstract languages for different purposes. It is also possible and useful, to transform one or more conceptual models (the source) to one or more different conceptual models (the target), where the target may be based on the same or different abstract languages from those of the source so that the target is of a different level of abstraction of the source, and has different representations of source concepts, provided certain conditions on the transformation are met. The most fundamental criterion for such a transformation is

that it is what mathematicians call ‘functional’. This means that any representation or particular combination of representations in the source model can have only one representation or combination of representations in the target model. Because of the uniqueness of the function, the meaning of the representation in the target model can be said to be well defined in terms of the source, i.e. each target representation can be said to represent a concept that is a specific function of certain source concepts. It is to be emphasized that the transforms spoken of are semantic, not merely symbolic. It is the representations of concepts that are transformed, not just symbol mapping. This may involve a deeper level of mapping of sets of concepts and not merely a concept-by-concept mapping. For example, if a vocabulary adopts a definition from another vocabulary, a high-fidelity semantic merge would require the adopter to adopt also the definitions of all concepts in the definition of an adopted concept, recursively. Thus, to achieve a high-fidelity merge, the transitive closure, or “deep adoption” of an definition must be used.

Not all representations in a source need be present in a target, and conversely. Information is usually filtered from the source or added in the target. In general, model transformations are not reversible. However, a chain of such transformations, the meaning of each correctly mapped representation can be interpreted in terms of an ultimate source conceptual model. If that ultimate source conceptual model is a model of the business domain in English or other natural language, it can be validated to have a well-defined business meaning by business personnel who are domain experts. It is often desirable to design transformations so the function preserves the semantics of the transformed representations individually. Sometimes it is desirable that the function be reversible, bidirectional. A rigorous mathematical account of the theory of model transformations in MDA remains to be developed. The new MDA specification for MOF Query, View, and Transformation (QVT) is the first step of the OMG to formalize transformations.

There is a special category of transformation in MDA that is noteworthy. These are the transformations between a conceptual model and a presentation of the model in graphical form. These transformations must be one-to-one. Each notational symbol corresponds to a concept in the conceptual schema of the conceptual model. Each symbol on a diagram of the conceptual model corresponds to a particular set of facts in the conceptual model that instantiate the concept represented by the symbol. Since alternative notations are available in some cases, the reverse mapping is not unique. This kind of mapping is called in mathematics an injective morphism. If the notation is unique, then the mapping is an isomorphism, also known as a bijective morphism.

#### Conclusion

An overview account of Model-Driven Architecture has been provided in terms of SBVR, giving also a demonstration of the expressivity and natural language modeling approach with SBVR. Language is the main medium of social and technical constructivism. A coherent and logically consistent account of any knowledge domain can be developed in natural language using SBVR. Such SBVR models can be subjected to MDA transformations. SBVR semantic formulations provide a basis in formal logic for analyzing and processing SBVR models. Availability of interoperable machine-readable representations of SBVR natural language models in MOF/XMI bridges the gap between the world of thought and the technology world. SBVR provides a natural starting point for the MDA process, by using the natural language that is the basis for thought and person-to-person communications for modeling what people have in mind.

To learn more about how you can use SBVR, contact Stan Hendryx, [stan@hendryxassoc.com](mailto:stan@hendryxassoc.com).

#